

## Front-end Frameworks

Module 1	▼
Module 2	▼
Module 3	▼
Module 4	▼
Module 5	▼
Course Assignment	▲

---

🕒 Course Assignment (../front-end-frameworks/ca.html)

---

🕒 Marking Criteria (../front-end-frameworks/marking-criteria.html)

# Course Assignment

### Table of Contents

- Goal
- Brief
- Process
- Delivery
- Resources

## Goal

To apply knowledge of React to build an eCom store.

## Brief

The API you are using for this brief is: <https://v2.api.noroff.dev/online-shop>

You can find individual items by appending a product ID at the end of the API URL e.g.  
<https://v2.api.noroff.dev/online-shop/f99cafd2-bd40-4694-8b33-a6052f36b435>

You are tasked with build out the following pages for an eCom store:

1. Homepage
2. Individual product page
3. Cart page
4. Checkout success page
5. Contact page

The Homepage should have a list of all the products. There should be a look-ahead search bar that filters products when typing in a product name. Clicking on a product should take a user to an individual product page.

You pages should use a `<Layout>` component that contains a header and footer. The header should contain a nav bar as well as a Cart icon component that acts as a button as well as displays the current number of items in the cart.

The individual product page should display data for a single product. There should be an `Add to cart` button which, upon clicking, adds the product to the cart. The product page should display the title of the product, the description and the image. There should also be reviews listed for the product, if there are any. You should use the `discountedPrice` property to display the price of the product. If there is a difference between the `discountedPrice` and `price` properties then that means there is a discount for that product. Calculate what this discount is and display it on the page.

Clicking on the Cart icon will load the Cart page, which will list all of the products as well as a total. The Cart page will have a Checkout button. Clicking this Checkout button then goes to a Checkout success page.

The Checkout success page will display a message to the user notifying them that their order was successful. There should also be a link that lets a user go back to the store. The cart must be cleared if the user gets to the Checkout success page.

There will be a contact page which will contain a contact form with the following fields. There must be form validation:

1. Full name (Minimum number of characters is 3, required)
2. Subject (Minimum number of characters is 3, required)
3. Email (Must be a valid email address, required)
4. Body (Minimum number of characters is 3, required)

You will be using React Router to switch between pages.

Your design should be responsive. You are welcome to use a CSS Framework, however, you're encouraged to design from scratch and use `styled-components` or CSS Modules.

You are not required to use TypeScript.

Your code is expected to be clean and well-formatted.

## Process

1. Create a new CRA app.
2. Create a Header that has a Nav.
3. Create a Cart Icon component and position this next to your Nav. This Cart Icon component will have an overlay that displays the number of items in the cart.
4. Create a Footer component.
5. Create a Layout component that has your Header and Footer.
6. Create the other pages:
  - 6.1 ContactPage
  - 6.2 ProductPage
  - 6.3 CheckoutPage

## 6.4 CheckoutSuccessPage

7. Add React Router and route to each of the pages. The ProductPage page will be using a dynamic segment.
8. Fetch the list of products on the Homepage and store this as a state.
9. On the homepage, loop through the products and display a Product component for each of the values. This Product component should look like a product card. Each Product component will have a `View product` button which will link to the ProductPage page.
10. The homepage should have a lookahead/auto-complete Search bar component. Typing values in the search bar should display products where the title matches the search input. Clicking on an item should take the user to the ProductPage page. Tip: Filter the user input and then display products that match the input.
11. On the ProductPage, use the ID of the product as the params for the dynamic segment. Add the product details as mentioned in the brief.
12. Create a cart state. When the `Add to cart` button on the ProductPage is clicked, add the product to the cart.
13. Clicking on the Cart Icon component will take the user to the CheckoutPage page.
14. The CheckoutPage must list all of the products in the cart, show a total at the bottom and a `Checkout` button.
15. Clicking the `Checkout` button will take the user to the CheckoutSuccessPage.
16. The CheckoutSuccessPage should display that the order was successful and clear the cart. There should be a link to go back to the store.
17. On the ContactPage, create the following inputs with the following requirements.
  - 16.1 Full name (Minimum number of characters is 3, required)
  - 16.2 Subject (Minimum number of characters is 3, required)
  - 16.3 Email (Must be a valid email address, required)
  - 16.4 Body (Minimum number of characters is 3, required)
  - 16.5 Submit button
18. `console.log` the data from the form once validation requirements are met.
19. Once your project is done, deploy it to Netlify.



## Delivery

Deploy your website on Netlify.

Please submit your public GitHub repository URL and the URL for your Netlify live site.

Please make sure to exclude `node_modules` by using a `.gitignore` file.

## Resources

- [Noroff API Docs](#) 
- [Using branches and making a pull request](#) 
- [Excluding node\\_modules using .gitignore](#) 